Utilisation d'outils de plongement d'Internet pour l'agrégation de ressources hétérogènes

Olivier Beaumont, Nicolas Bonichon, Philippe Duchon, Hubert Larchevêque

Utilisation d'outils de plongement d'Internet pour l'agrégation de ressources hétérogènes[†]

Olivier Beaumont[1] and Nicolas Bonichon[2] and Philippe Duchon[2] and Hubert Larchevêque[3]

[1]*INRIA Bordeaux Sud-Ouest.* [2]*Université de Bordeaux.* [3]*Institut Polytechnique de Bordeaux*

Dans cet article, nous nous intéressons aux plates-formes de grande échelle comme BOINC, qui sont constituées d'un ensemble de ressources hétérogènes utilisant Internet comme réseau de communication. Dans ce contexte, nous étudions un problème d'agrégation de ressources dans lequel l'objectif est de construire des groupes de ressources, de telle sorte que chaque groupe ait une capacité totale supérieure à une certaine valeur, et tels qu'au sein d'un même groupe, deux ressources ne soient pas trop éloignées (en terme de latence) l'une de l'autre. Dans de telles plateformes, il n'est pas réaliste de supposer connaître la latence pour l'intégralité des couples de noeuds. Il est donc nécessaire d'avoir recours à des outils de plongement comme Vivaldi ou Sequoia. Ces outils permettent de travailler dans des espaces métriques spécifiques et dans lesquels la distance entre deux noeuds peut être obtenue directement à partir d'une petite quantité d'informations disponible à chaque noeud.

Nous étudions le problème *Bin Covering* avec Contrainte de Distance (BCCD) et utilisons des algorithmes dédiés dans les espaces métriques induits par plusieurs outils de plongement pour proposer une comparaison de ces algorithmes en nous appuyant sur des mesures de latences réelles. Cette comparaison nous permet de décider quel couple (algorithme,outil de plongement) est en pratique le plus efficace pour ce problème d'agrégation de ressources.

Keywords: Outils de plongement d'Internet, agrégation de ressources, algorithmes d'approximation

1 Introduction

Dans cet article nous étudions l'utilisation de deux outils de plongement d'Internet pour l'agrégation de ressources dans des plateformes de grande échelle. Les outils de plongement d'Internet sont utilisés pour placer dans un espace (généralement métrique) simple des ressources distribuées connectées via Internet. Vivaldi [4] et Sequoia [7] sont de tels outils de plongement parmi les plus rencontrés dans la littérature. Ces outils assignent à chaque ressource (noeud) une position dans un espace simple, de telle sorte que la distance entre n'importe quelle paire de noeuds dans cette espace soit une bonne approximation de la distance réélle les séparant (leur latence) dans Internet. Une alternative évidente pourrait être de mesurer toutes les latences et de travailler sur la matrice de latence complète. Dans le contexte de plateformes à grande échelle qui nous intéresse, cette approche n'est pas réaliste, en raison du coût nécessaire pour effectuer ces mesures, et du dynamisme intrinsèque à ce type de plateformes.

L'utilisation d'outils de plongement provoque une (légère) distorsion des latences, mais en contre-partie permet de travailler dans des espaces plus simples, et ainsi d'élaborer des algorithmes d'approximation efficaces. Dans le cadre d'une application spécifique, comme l'agrégation de ressources qui nous intéresse, seule la performance du couple (plongement,algorithme d'approximation) est importante.

Dans cet article, nous considérons 3 plongements : le plongement identité (*i.e.* utilisation directe de la matrice de latence), Vivaldi et Sequoia. Pour chacun de ces plongements, nous considérons un algorithme d'approximation spécifique, qui utilise les propriétés structurelles de l'espace auquel il est dédié.

Nous comparons alors les performances des différents couples (plongement, algorithme d'approximation) en utilisant un jeu de données s'appuyant sur des mesures de latences effectuées sur PlanetLab [‡].

[†] Le contenu de cet article est tiré de [3]

[‡] http ://www.planet-lab.org/

Bin Covering [1] est un problème classique d'optimisation combinatoire. Dans cet article nous nous intéressons au cas où nous disposons d'un ensemble d'éléments (les ressources) pondérés (hétérogènes) S et des distances (latences) entre chaque paire d'éléments. L'objectif est de regrouper ces éléments en un maximum de groupes agrégeant chacun un poids minimal. Ici, nous nous intéressons à une généralisation appelée *Bin Covering* avec Contrainte de Distance (BCCD), en y introduisant une contrainte de distance : la distance maximale entre deux éléments d'un même groupe ne doit pas dépasser une certaine valeur seuil $d_{\max}$. Théoriquement, dans cette généralisation, les éléments sont placés dans un espace semi-métrique (*i.e.* la notion de distance ne vérifie pas nécessairement l'inégalité triangulaire). Dans BCCD, l'objectif est de construire un maximum de groupes de diamètre au plus $d_{\max}$ et de poids supérieur à une valeur seuil W.

Des motivations pour l'étude de BCCD (qu'on peut trouver dans [2], où ce problème a déjà été étudié) concernent des plateformes hautement distribuées, comme BOINC [§] ou Folding@home [¶]. Actuellement toutes les applications utilisant ce type de plateformes sont constituées d'un très grand nombre de tâches indépendantes, et toutes les données nécessaires au traitement d'une tâche doivent être stockées localement sur le noeud qui la traite. Nous considérons le cas où l'ensemble des données nécessaire au traitement d'une tâche est trop volumineux pour être stocké sur un seul noeud. Dans ce cas, les besoins en stockage et en ressources de calcul doivent être répartis sur un ensemble de noeuds qui va collaborer pour traiter une même tâche. Les noeuds présents au sein d'un même groupe de travail doivent alors avoir une capacité agrégée (en terme de mémoire, de puissance de calcul, etc.) suffisante, plus grande qu'une certaine valeur seuil, et on souhaite qu'ils soient proches les uns des autres (que la latence entre chaque paire de noeuds soit raisonnable) de façon à éviter des latences importantes durant les communications au sein d'un même groupe de travail. Construire de tels groupes de travail revient à résoudre BCCD, en utilisant les latences comme distance entre les noeuds.

Dans cet article nous considérons des algorithmes d'approximation avec augmentation de ressource. Plus précisément, un algorithme $\mathcal{A}$ est un (α,β)-algorithme d'approximation pour BCCD si il s'exécute en temps polynomial et construit des groupes de diamètre au plus $\beta d_{\max}$. Le nombre de groupes qu'il construit est au moins $\alpha \textsc{Opt}^*$, avec $\alpha \leq 1$, où $\textsc{Opt}^*$ est le nombre de groupes d'une solution optimale contrainte à construire des groupes de diamètre $d_{\max}$. Remarquons que le recours à l'augmentation de ressource est nécessaire, car pour $\beta = (2-\varepsilon)$, BCCD reste inapproximable à facteur constant dans le cas où les points sont plongés dans un espace métrique. Ceci se montre via une réduction au problème MAXIMUM CLIQUE.

Dans la suite de l'article nous présentons les outils de plongements ainsi que les espaces considérés. Ensuite nous présentons des algorithmes d'approximation adaptés à ces différents plongements. Enfin nous présentons l'étude expérimentale des couples (plongement, algorithme).

2 Espaces de plongements

Dans cette section, nous présentons certaines caractéristiques de l'espace des latences d'Internet et deux des outils de plongement les plus répandus dans la littérature : Vivaldi et Sequoia.

L'espace des latences d'Internet n'est pas un espace métrique, mais seulement un espace semi-métrique, car il contient un grand nombre de violations de l'inégalité triangulaire [6]. Ces violations d'inégalités triangulaires viennent par exemple de la charge du réseau ou de la configuration des routeurs qui font que le plus court chemin n'est pas forcément celui emprunté par les paquets. Cependant, comme le montre ces travaux [5], l'espace des latences d'Internet semble être raisonnablement proche d'une ρ-infra-métrique un espace dans lequel l'inégalité triangulaire est relachée de la façon suivante : $d(u,v) \leq \rho.\max(d(u,w),d(v,w))$ pour tout triplet de noeuds (u,v,w). Plus précisément les résultats [5] montrent que 99.5% des triplets vérifient cette inégalité pour $\rho = 2$.

L'espace des latences d'Internet n'étant pas métrique, il ne peut être plongé dans un espace métrique sans subir de perte d'informations. Sequoia et Vivaldi permettent cependant une bonne approximation des latences entre chaque paire de noeuds [4].

Vivaldi, dans sa version standard, associe à chaque noeud d'un réseau deux coordonnées dans un plan Euclidien, plus une hauteur. Dans un tel espace, la distance entre deux noeuds de coordonnées $(a_x,a_y,a_h) \in$

[§] http ://boinc.berkeley.edu/
[¶] http ://folding.stanford.edu

$\mathbb{R}^2 \times \mathbb{R}^+$ et $(b_x, b_y, b_h) \in \mathbb{R}^2 \times \mathbb{R}^+$ est : $d(a,b) = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2} + a_h + b_h$. Dans la suite nous appellons cet espace métrique l'espace *Vivaldi*.

Sequoia plonge les noeuds d'Internet dans un ou plusieurs arbres pondérés dans lesquels chaque noeud est soit une feuille soit une racine, et dans lesquels les noeuds internes sont des noeuds virtuels. La distance entre deux points dans le réseau d'origine est bien approchée par la distance dans l'arbre de plongement (ou par la médiane des distances dans le cas de plusieurs arbres). Comme présenté dans [7], la précision des prédictions obtenues en utilisant Sequoia est meilleure dès que quelques arbres sont utilisés. Dans la suite nous travaillons avec un seul arbre.

3 Algorithmes d'approximation de BCCD

Nous présentons ici trois algorithmes d'approximation de BCCD fonctionnant sur des espaces métriques différents. Ces algorithmes et leurs preuves sont décrits en détails dans [3].

Le premier algorithme (*Algorrthme 1*) considère que les noeuds sont dans un espace ρ-infra-métrique. Cet algorithme fonctionne en deux phases. Lors de la première, on considère toutes les boules de rayon $d_{\max}$. Dans chacune de ces boules, on applique l'algorithme Next-Fit-Decreasing (NFD) sur les éléments de la boule. Cet algorithme trie les noeuds par ordre décroissant, puis les place dans la bin courante. Lorsque le poids de la bin courante a atteint la valeur seuil, une nouvelle bin est ouverte et devient la bin courante. La seconde phase est similaire à la première, sauf qu'elle travaille sur des boules de rayon $\rho d_{\max}$.

L'*Algorithme 2* travaille dans un espace Vivaldi et fonctionne également en deux phases au cours desquelles l'algorithme NFD est appliqué sur des sous-ensembles de noeuds appelés *lentilles*. Durant la première phase, un nombre quadratique de lentilles de diamètre au plus $\frac{\sqrt{3}+1}{2}.d_{\max}$ est examiné. Dans la seconde phase, un nombre quadratique de lentilles de diamètre au plus $(2+\frac{3\sqrt{3}}{2})d_{\max}$ est examiné.

Enfin l'*Algorithme 3* travaille sur un ensemble de points plongés dans un arbre enraciné. Cet algorithme repose sur un parcours post-fixe de l'arbre. A chaque étape de ce parcours, on applique l'algorithme NFD sur le sous-arbre enraciné sur le noeud courant et de profondeur au plus $d_{\max}$.

Théorème 1 *L'Algorithme 1 est* $(2/5, \rho^2)$*-algorithme d'approximation pour BCCD dans une* ρ*-infra-métrique.*

L'Algorithme 2 est $(\frac{2}{5}, 2+\frac{3\sqrt{3}}{2})$*-algorithme d'approximation pour BCCD dans un espace Vivaldi.*

L'Algorithme 3 est $(\frac{1}{3}, 2)$*-algorithme d'approximation pour BCCD dans une métrique d'arbre.*

Une rapide analyse montre que les algorithmes 1,2 et 3 ont respectivement une complexité en $O(n^2), O(n^{4,5})$ et $O(n^2)$.

4 Simulations

Nous utilisons un jeu de données réel pour estimer les latences, issu du projet Meridian [‖], contenant toutes les mesures de latence relatives à un ensemble de 2500 noeuds choisis arbitrairement sur la plateforme PlanetLab. Pour estimer les capacités hétérogènes des ressources, nous nous appuyons sur un jeu de données du projet XtremLab [**]. XtremLab consiste en une liste des caractéristiques des noeuds utilisés par des applications BOINC. Nous en extrayons les données correspondant aux capacités de calcul mises à disposition par chaque noeud sur la plateforme (leur valeur en FLOPS), obtenant une puissance totale de $3632 GFlops$.

Partant de ces données nous effectuons le plongement de la matrice de latence en utilisant soit Vivaldi soit Sequoia (avec seulement un arbre pour Sequoia). Nous appliquons à chacun de ces plongements l'algorithme correspondant pour BCCD, le tout sur le même ensemble d'éléments pondérés, et pour différentes valeurs de $d_{\max}$, allant de la latence minimale entre deux points, préalablement identifiée, à une valeur de $200ms$ (au delà de cette valeur, aucune évolution n'est observée).

Nous avons effectué des simulations pour différentes valeurs de W, et avons choisi $W \simeq 11 GFlops$ comme étant la valeur la plus représentative (chaque groupe doit donc agréger environ 0.3% du poids total). Pour de plus petites valeurs de W, les solutions tendent à placer de moins en moins d'éléments dans chaque

[‖] http ://www.cs.cornell.edu/People/egs/meridian/data.php

[**] http ://xw01.lri.fr :4320/

groupe, puisque les poids des éléments sont plus proches de la valeur seuil. Le nombre de groupes construits tend alors vers une valeur stable pour une plus petite valeur de $d_{\max}$. En utilisant de plus petites valeurs pour W, la contrainte de poids devient la plus importante et BCCD se rapproche d'un problème de *Bin Covering* classique, puisque la contrainte de distance est de moins en moins sensible : même quand $d_{\max}$ augmente, la construction de groupes peut toujours s'avérer difficile en raison du manque de poids. En utilisant cette valeur de W, un groupe contient en moyenne environ 7 éléments.

La Figure 1 présente à gauche le nombre de groupes valides, et à droite le diamètre moyen des groupes construits pour différentes valeurs de $d_{\max}$, en utilisant chacune des combinaisons (plongement,algorithme dédié) que nous avons sélectionnés. Dans tous les cas, l'Algorithme 1 appliqué à la matrice de latence offre

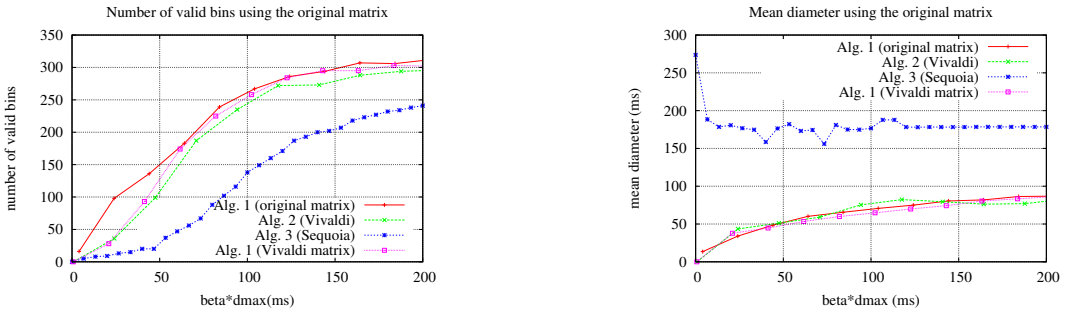

FIG. 1: Nombre de groupes valides (gauche) et diamètre moyen des groupes construits (droite) pour BCCD, pour différentes valeurs de $d_{\max}$, en utilisant différents outils de plongement.

de meilleures performances à la fois que l'Algorithme 2 et que l'Algorithme 1 utilisant la matrice de latence obtenue à partir du plongement dans un espace Vivaldi. Tous ces algorithmes ont de meilleurs performances que l'Algorithme 3.

L'Algorithme 2 et l'Algorithme 1 utilisant la matrice de latence obtenue à partir du plongement dans Vivaldi ont des performances très comparables. Il est intéressant de constater que, bien que l'Algorithme 2 requiert une augmentation de ressources de $\beta \simeq 4.6$, le diamètre moyen des groupes qu'il construit tend à être équivalent au diamètre moyen des groupes construits par l'Algorithme 1, qui lui requiert seulement une augmentation de ressources de $\beta = 4$.

L'utilisation de Vivaldi offre des performances quasi équivalentes à l'algorithme dédié au cas général, malgré la perte de précision dans la prédiction des distances induite par le plongement. Ainsi cette perte de précision due au plongement semble bien compensée par l'élaboration d'un algorithme efficace dédié à un espace métrique plus simple. Par ailleurs, cela montre qu'utiliser un outil de plongement conjointement à un algorithme dédié à celui-ci permet d'obtenir des résultats aussi bon qu'en utilisant une matrice de latence complète, alors qu'obtenir une telle matrice de latence est, en pratique, impossible.

Références

[1] S.F. Assmann, D.S. Johnson, D.J. Kleitman, and J.Y.T. Leung. On a dual version of the one-dimensional bin packing problem. *Journal of algorithms*, 5(4) :502–525, 1984.

[2] O. Beaumont, N. Bonichon, Ph. Duchon, and H. Larcheveque. Distributed approximation algorithm for resource clustering. In *OPODIS 2008*, volume 5401 of *Lecture Notes in Computer Science*, pages 564–567. Springer, 2008.

[3] Olivier Beaumont, Nicolas Bonichon, Philippe Duchon, and Hubert Larchevêque. Use of Internet Embedding Tools for Heterogeneous Resources Aggregation. Research report. RR INRIA-00562655 (accepté à HCW 2011).

[4] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi : a decentralized network coordinate system. *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 15–26, 2004.

[5] Emmanuelle Lebhar, Pierre Fraigniaud, and Laurent Viennot. The inframetric model for the internet. In *Proceedings of the 27th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1085–1093, Phoenix, 2008.

[6] Cristian Lumezanu, Randy Baden, Neil Spring, and Bobby Bhattacharjee. Triangle inequality variations in the internet. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, IMC '09, pages 177–183, New York, NY, USA, 2009. ACM.

[7] Venugopalan Ramasubramanian, Dahlia Malkhi, Fabian Kuhn, Mahesh Balakrishnan, Archit Gupta, and Aditya Akella. On the treeness of internet latency and bandwidth. In *SIGMETRICS '09 : Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems*, pages 61–72, New York, NY, USA, 2009. ACM.